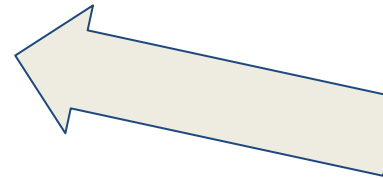


StyleCLIP - Project Outline

Alex Leonardi, Ben Harpe, Michael Fein

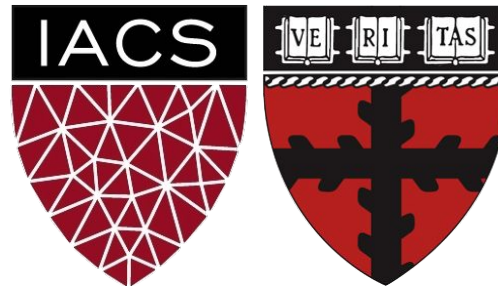
AC295



AC215

Pavlos Protopapas

Institute for Applied Computational Science, Harvard



Problem Overview

Problem Definition

Pictures of various people exist but are normally pretty boring and we have little control over them. If you want to edit pictures in exciting ways, you have to learn Photoshop, which is quite difficult. There is no easy way to apply transformations to people in images and we'd like to be able to do so without much time or computational power.

Proposed Solution

We'll have an app on our phone that allows us to take or upload pictures. We will then be able to type in a description of the current picture and a description of the transformation we'd like to make. The app will then make that transformation to the image and return it. For example, we could take a picture of Pavlos and caption it "a face." If we added the transformation description "a bald face," nothing would happen.

Project Scope



Proof Of Concept (POC)

- Be able to run pretrained StyleCLIP model on our images
- Find additional images to train on (Celeb_a_HQ dataset)
- Finetune e4e model (used for StyleCLIP global) on new images
- Connect new model to CLIP and be able to generate new transformations

Prototype

- Create a mockup of screens to see how the app could look like
- Deploy StyleCLIP with finetuned e4e to Fast API to service model predictions as an API

Minimum Viable Product (MVP)

- Create containerized app to apply transformations to images
- API Server for uploading images and predicting using best model

What is StyleCLIP?

StyleCLIP Overview

- StyleCLIP is a combination of machine learning models that allows users to transform an input image based on an input text
- StyleCLIP works by combining the generative power of StyleGAN with the text-based image manipulation provided by OpenAI's CLIP




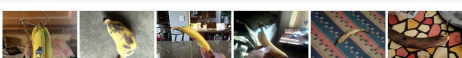
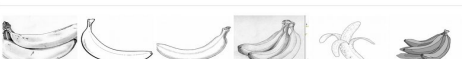
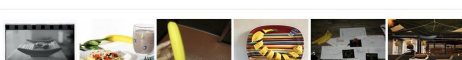


StyleGAN and CLIP

- Allows users to generate images that match a given set of attributes
- Accomplishes this output using its latent space, which allows for disentanglement of various factors such that individual factors can be adjusted

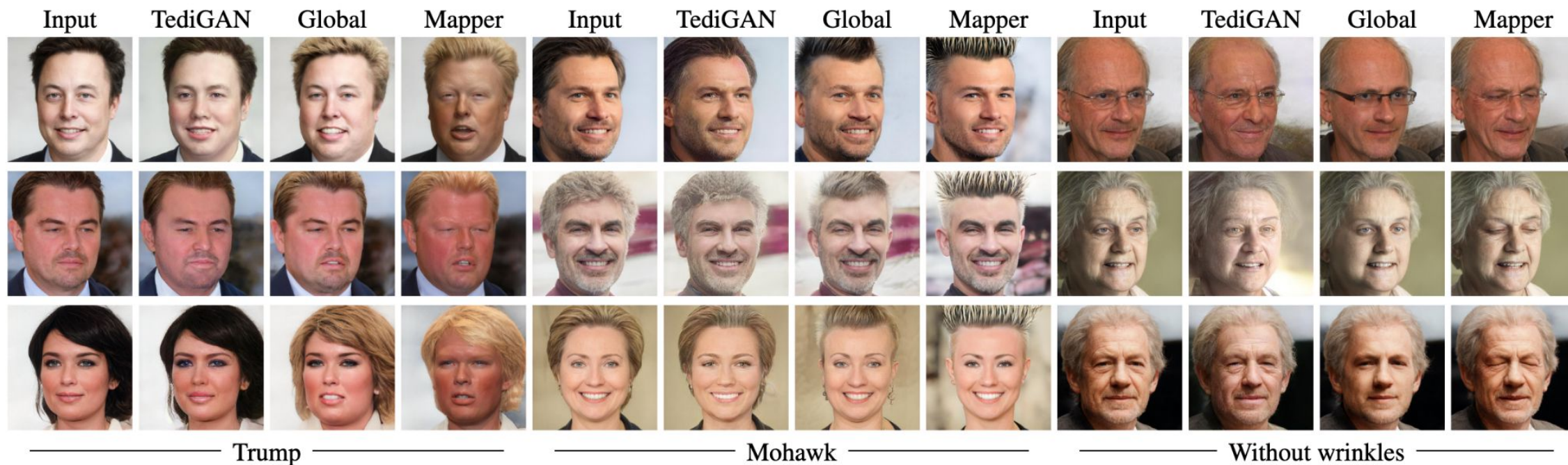


- Developed to generalize image classification to a larger variety of benchmarks by not directly optimizing for specific benchmarks
- Achieved the ability to effectively associate images with their names

DATASET	IMAGENET RESNET101	CLIP VIT-L
 ImageNet	76.2%	76.2%
 ImageNet V2	64.3%	70.1%
 ImageNet Rendition	37.7%	88.9%
 ObjectNet	32.6%	72.3%
 ImageNet Sketch	25.2%	60.2%
 ImageNet Adversarial	2.7%	77.1%

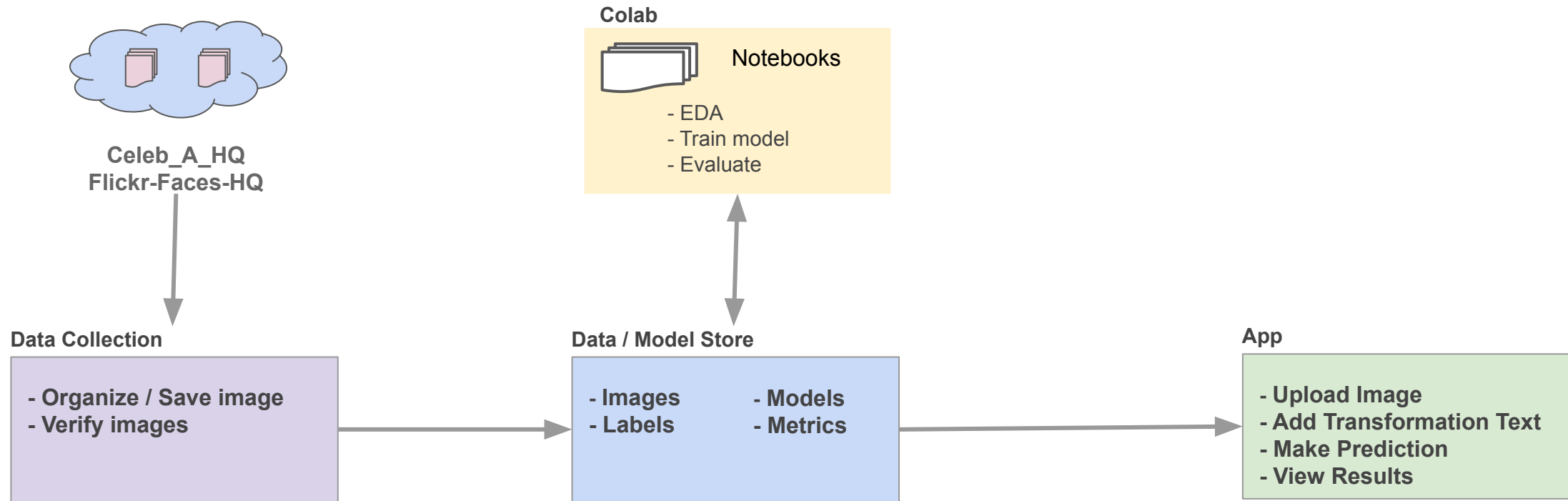
StyleCLIP Methodologies

- StyleCLIP combines StyleGAN and CLIP using three different methods
- The first two methods described in the paper require training a new model for every individual image, so for our model, we focus on the Global Directions method, which can be quickly run on a new input image



Our Architecture

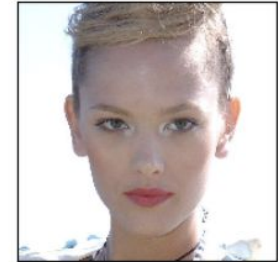
Process Flow



Our Model

Data - celeb_a_hq

- Total number of images: 30000
- Dimensions: (1024,1024)
- Attributes: 40 binary, 5 landmarks



Model - e4e

1. Initialize encoder weights to pretrained model
 2. Freeze all layers except Encoder/Styles
 3. Train model on CelebA dataset for 10,000 iterations
 4. Choose preferred output model
- Encoder
 - Input Layer
 - Body
 - Styles
 - Latent Layers
 - Decoder
 - Style
 - Input Layer
 - Convolutional Layers
 - To RGB
 - Noise
 - Face Pool

Models - Training Results

- Though the model was able to train on CelebA and eventually produce a decent output, it performed worse than normal StyleGAN

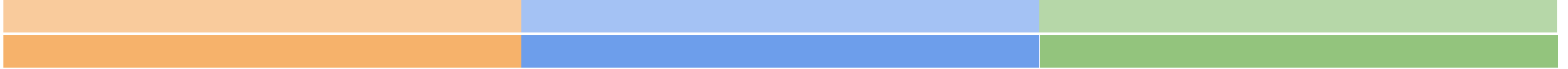
Initial Training (All Layers Trained)



Final Training (Most Layers Frozen)



Challenges



Weak Performance

- The models trained on CelebA generally had weaker performance than the pretrained models
- We were able to combat this problem to some extent by freezing layers and tuning hyperparameters
- Even with these changes, the performance was still not as good, demonstrating the sensitivity of the model

Layer Freezing

- Initial attempts to freeze layers did not work because of the relatively complex architecture of the model
- We were eventually able to freeze all layers except for the Styles layers in the encoder, but choosing which layers to freeze was a difficult task
- In the end, we chose to freeze all layers except for the Styles layers to preserve as much of the initial encoding as possible

Memory Limitations

- The original encoding models were trained on more powerful machines, which allowed for the use of larger batch sizes
- Some of our weak performance could likely be attributed to the use of very small batches necessitated by our available GPUs